SCS 2201 - Data Structures and Algorithms III

Greedy Algorithms

Group Assignment

Report:

Greedy components in each question and the approach taken to solve it

<u>Group 17 - Members</u>

- **18000444: DISSANAYAKA J.R.D.D.G.**
- **18000452: DISSANAYAKE D.M.P.M.**
- **18000462: DODAMPE A.H.**

Question 01

Candidate set:

Barrels in the wine yard

Initialised an array to include the volume of each barrel in bottles.

Selection Function:

From the given set of barrels in the wine yard, the barrel with the highest value density (price per bottle) is taken as the best choice where sandun can start filling the bottles he has brought.
Price per bottle in each barrel was calculated by dividing each element of the array containing price of each barrel from the corresponding element of the array containing the volume of barrels (in bottles).
Then the barrel with the highest price per bottle was sorted out.

Feasibility function:

First Sandun can fill the bottles he has got, from the barrel with highest price per bottle. If he gets more bottles remaining to be filled,
he can fill them from the barrel with the next highest price per bottle. This process is followed until every bottle is filled.
As said, we assure that sandun won't fill any of the bottles with more than one type of wine.

Objective Function:

Once the barrel with highest price per bottle is sorted out, Sandun can fill the bottles he got starting from that particular barrel.

CONTINUED…

Pseudo code:

```
maxValWine(new_bottles_arrange[i..n],
newval_barrel[i..n],bottles_sandun_brought, number_of_barrels) :
SET i =0
SET bag = 0
while(bag!=bottles_sandun_brought and i!=number_of_barrels)
 i = index of best price per bottle
      If(bag + new_bottles_arrange[i]<=bottles_sandun_brought):
           bag += new_bottles_arrange[i]
           maxVal += newval_barrel[i]*new_bottles_arrange[i]


      else
            maxval += (bottles_sandun_brought-bag)*newval_barrel[i];
            bag += (bottles_sandun_brought-bag);
      end if
 i++
end while
```

Solution Function:

Once the Sandun's bottles are filled from the barrels descending from highest price per bottle, Maximum value of wine that Sandun gets from this deal can be determined.

Question 02

Candidate Set:

Students of the class.

Initialised an array based on the seated positions of the students and their scores are considered for selection.

Selection Function:

Considering two students seated next to each other and comparing their performance based on the scores obtained. One with the highest mark will get more masks.

Feasibility Function:

As it mentioned in the question, every student must get at least one mask so every student will add at least one unit(masks) to the solution.

Objective Function:

As the teacher wants to buy minimum number of masks, first two students will get one or two masks according to their performance*. Afterwards if the next student is performance is better than the previous one (since only considering two students seated next to each-other each time) he/she will one more mask than the previous one. If equal no change in number of masks. If less will only get one mask*(in order to minimise the number of masks must be bought).

Pseudo Code:

```
SET total_masks = 0
 while( x < number of students):
      if(first two students):
           student with the higher mark -> 2 masks
           student with the lesser mark -> 1 mask
           total_masks += 3
      else if(not first two but performance is better than previous):
           student gets -> previous student's masks+1
           total_masks += previous student's masks+1
```

CONTINUED…

```
else if(not first two but performance is lesser than previous):
    student gets -> 1 mask
    total_masks += 1
else //performance equal
    student gets -> previous student’s masks
    total_masks += previous student’s masks
```

Solution Function:

Once all the students have been considered the number of the masks that should be bought is calculated. It will get printed.

* **:** As a greedy technique is used the number of masks which are selected are local optimal solutions which would eventually lead to a global optimal solution. Global solution may not be the most optimal solution.

Question 03

Candidate Set:

The Products which are ready to ship.

Initialised an array to contain the candidate set.

Selection Function:

The products that are ready to ship must be categorise as a container can only contain products that are within the range of "*minimum weight to minimum weight + 4 units*"

Sorted the candidate set in ascending order to get the minimum values first.
Selected the first minimum weighted product as the best fit.

Feasibility Function:

Since a minimum weighted product has been already chosen wrote a function to determine the feasibility of the remaining candidates to go into the container. If the a product exceeds the limit of weight it will be put into another container which will be done in the selection function.

Objective Function:

Once a minimum value is chosen a container is reserved for it. Another container will be allocated once a product is found, weight is greater *minimum weight + 4*

Pseudo Code:

```
SET temp_val = 0
while( x < number of products):
        if(weight[x](sorted weights in Ascending) > temp_val):
                temp_val = weight[x] + 4
                number of containers ++
        end if
end while
```

Solution Function:

Once all the candidates' feasibilities have been checked, number of containers that should be allocated have been determined. The determined value will get printed.